Dendritic Meshing*

LA-UR 11-04075

B. A. Jean, R. W. Douglass, G. R. McNamara, and F. A. Ortega

Los Alamos National Laboratory
MS T085
P.O. Box 1663
Los Alamos, NM 87545
baj@lanl.gov

1 Introduction

A mesh is said to be dendritic if it contains elements with mid-side (edge) nodes when the predominant element topology has only corner nodes. A dendritic mesh is illustrated in Figure 1 where the predominant element is a four-node quadrilateral, but has also several five-node quadrilateral elements each with one mid-edge node plus four corner nodes. Such meshes arise when an approximately uniform element size is required across a mesh domain in cases, for example, where domain geometry changes would otherwise cause a significant variation in element size or in an Adaptive Mesh Refinement (AMR) context. In meshes created for multi-physics applications with explicit time-stepping, the maximum time-step size is intimately tied to element size through the Courant-Friedrichs-Lewy condition[6]:

$$\frac{u\,\Delta t}{\Delta x} \leq C \tag{1}$$

where u is a representative speed, Δx the element size, Δt is the time-step size, and C is a constant appropriate for the physics being modeled. This implies $\Delta t \leq C'\Delta x$, C' a constant, and the smaller the element, the smaller the time-step size. The smallest element in a mesh therefore limits the time-step size providing motivation to equalize element size over a domain.

An additional concern is that the mesh adhere as nearly as possible to the domain boundary geometry. There are two general classes of boundary-fitted mesh generation methods (*e.g.,* [14]): block-structured and unstructured. Both of these methods have inherent strengths and weaknesses. The various unstructured methods are highly automated and tend to produce meshes with

*This work was performed at Los Alamos National Laboratory under the auspices of U.S. Department of Energy, under contract DE-AC52-06NA25396 and has been reviewed for general release as LA-UR 11-04075.

relatively uniform zone size. However unstructured techniques generally produce meshes with irregular zone connectivity and there is little or no control over zone orientation or zone aspect ratio. Structured methods enable control of zone orientation, have regular connectivity, and can produce very high aspect ratio zones (desired in some problem domains). However structured techniques are labor-intensive and control of zone-size is limited by the geometry of the problem and the domain decomposition into logical blocks. The discussion herein focuses on boundary-fitted meshes with a relatively unform zone size in most regions, the option to produce high aspect ratio zones in other regions, and the ability to align the mesh with the predominant direction of shock propogation and/or material flow. Neither of the existing meshing methods mentioned above satisfies these criteria.

To meet these meshing requirements, a hybrid method is presented which is termed *dendritc meshing*. Dendritic meshing is a modified block-structured technique that allows logical edges within a structured mesh block to be "deactivated" as needed to control zone size. This method combines many of the advantages of both structured and unstructured methods. With dendritic meshing, zone size can be kept relatively uniform when needed, zone aspect ratio can be controlled, zone orientation can be controlled, and nodes of irregular connectivity within the mesh are minimized.

The starting point for a dendritic mesh is a standard block-structured mesh; a mesh consisting of perhaps multiple blocks each having a logical ij-structure. To produce a dendritic mesh, selected segments of constant-i or constant-j lines are removed. The "dendrites" are the nodes at which an i or j line segment becomes inactive. After removing the selected mesh segments, the mesh is re-interpolated using a specialized dendrite-aware transfinite interpolation algorithm, and is then smoothed with a dendrite-aware smoother. Figures 1 and 2 show the active and removed edges of a structured mesh in physical and logical space, respectively.

The following sections highlight the significant issues in the dendritic meshing process. These issues are: data structures, feathering of a structured mesh, how to build the initial mesh, trans-finite interpolation (TFI) with dendrites, smoothing of dendritic meshes, and a brief conclusion.

2 Data Structures

The mesh is composed of a set of logically rectangular mesh blocks with the individual blocks connected together in an unstructured fashion[14] creating a block-structured mesh data structure. Mesh nodes for the entire problem are stored in a single master node array. The array index of a node in this master array is its *id*. The array of nodes for a mesh block stores the id's of the nodes for the block and not the nodes themselves. Block-to-block connectivity is determined implicitly by shared nodes, not by an explicit block topology data structure.

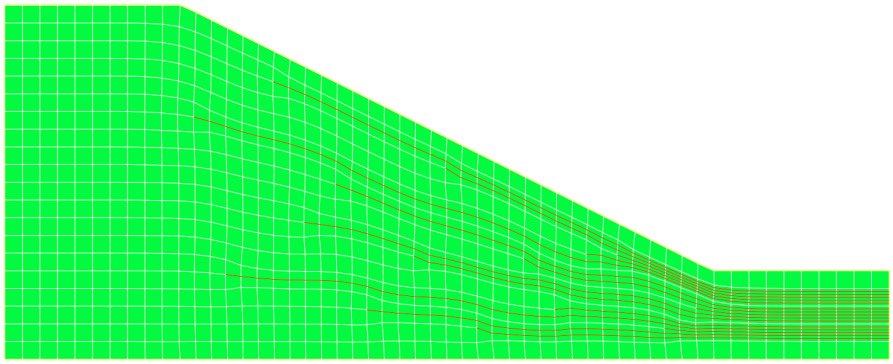


Fig. 1. A simple dendritic mesh with inactive edges shown in red.

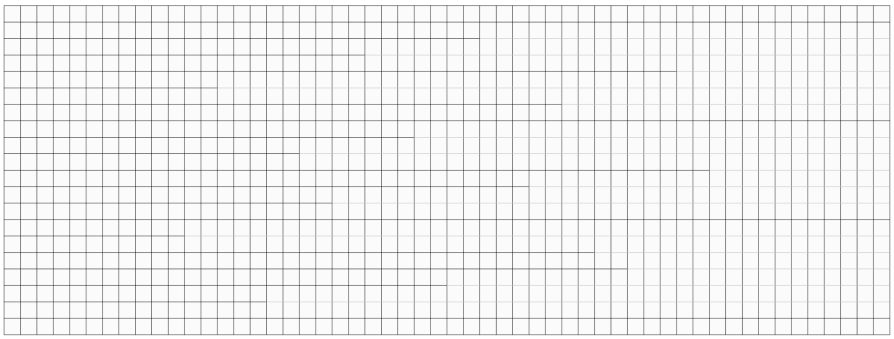


Fig. 2. Dendritic Mesh in Logical Space. The logical coordinate system origin is in the lower-left corner. The i-axis is horizontal and the j-axis is vertical.

Active Mesh Lines and Vertices

In addition to the array of mesh node id's, each block also maintains an array of Boolean ij-edge flags which indicate whether a zone mesh edge is active or inactive. A zone is identified by its lower left corner node indices, i, j. Figure 3 shows the edge flags associated with node (i, j).

Activity of a node is determined by whether or not its edges are active. A node is active if and only if at least one active i-edge and at least one active j-edge connect to the node. Figure 4 shows an inactive edge (i, j) and the associated inactive node at $(i + 1, j)$. The $(i + 1, j)$ node is inactive because no active i-edges touch it.

Node Slaving

Dendritic nodes (nodes at which ij-lines terminate) are also called *slave* nodes. Slave nodes are constrained to lie at the mid-point of the line connecting the slave's neighbors in the perpendicular logical direction of the terminating line.

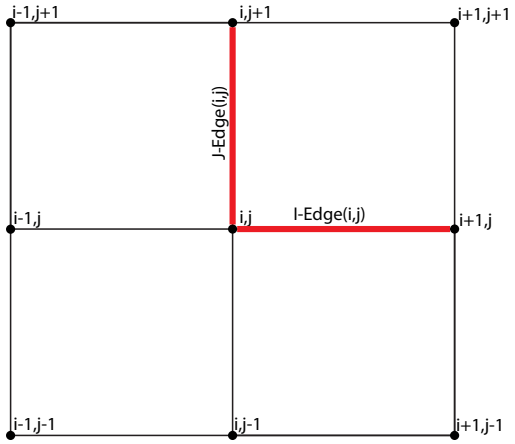


Fig. 3. Edge flags for node (i, j).

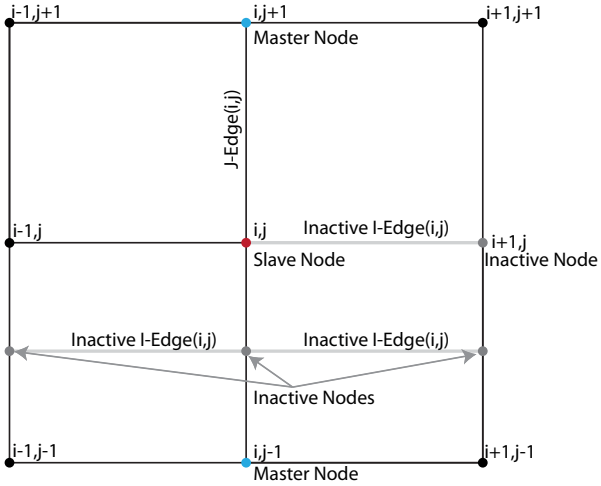


Fig. 4. Inactive edges/nodes and node slaving.

The slaving constraint for nodes on terminating i-lines and terminating j-lines are given in Equations 2 and 3, respectively.

$$\mathbf{r}_{i,j} = \frac{1}{2}\left(\mathbf{r}_{i,j+1} + \mathbf{r}_{i,j-1}\right) \tag{2}$$

$$\mathbf{r}_{i,j} = \frac{1}{2}\left(\mathbf{r}_{i+1,j} + \mathbf{r}_{i-1,j}\right) \tag{3}$$

Note that in the above equations, the $\pm$ indexing on the nodes denotes the next/previous *active* node in the indicated direction and not simply the adjacent node in the original structured mesh (see Figure 4).

2.1 Conversion to an Unstructured Mesh

An unstructured representation of the mesh is used for smoothing (see section 5) and for output to some physics codes. The unstructured mesh is defined as a collection of arbitrary polyhedral zones and is built by generating polyhedral zones from the active nodes. This is done by marching around the active edges in the block structured mesh where a change in direction is made when the next active edge in the opposite logical direction, for example when changing from an i-edge to a j-edge. When a change in direction is found, the active vertex at that change then becomes a corner vertex of the unstructured zone. Any vertices in the zone that is not a corner vertex then becomes a slave vertex and the two corner vertices on each side of the save vertex are the master vertices for that slave.

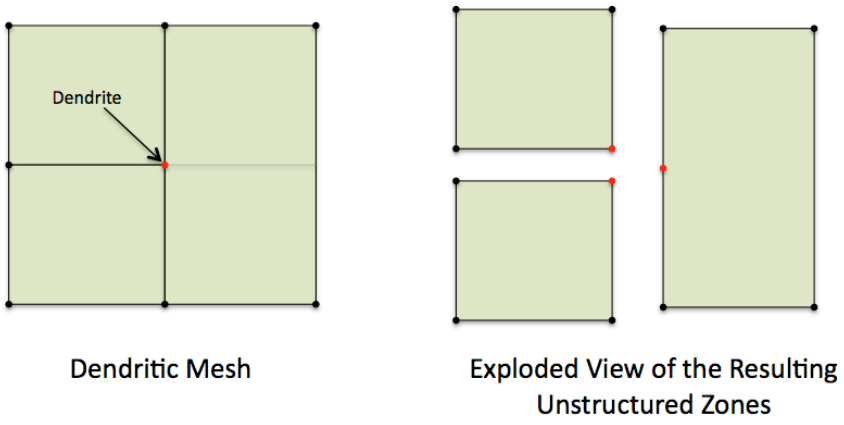


Fig. 5. Dendrite treatment during conversion to an unstructured mesh.

2.1.1 Arbitrary Polyhedral Topology

Due to the hanging vertices produced by dendritic mesh lines, an arbitrary polyhedral mesh topology is used for the unstructured mesh. This topology defines a zone by its faces and faces are defined by their vertices. The vertices in the face are ordered such that they define a right-handed normal relative to the zone center. In 2-D space, a face is an edge and the order of the vertices for each edges and the order of the edges are specified such that they define a right-handed normal relative to the X-Y axis plane. Because of the right-handedness of the arbitrary polyhedral topology, each zone has its own unique set of faces. Every polygonal zone in the mesh has it's own set of unique faces, however the faces between two zones have the same vertices but with a different sense. To provide mesh connectivity, the two faces with the same vertices are linked in the data structure.

2.2 Mesh Connectivity

Special consideration must be given to boundary nodes of connected dendritic mesh blocks. Figure 6 illustrates the cases that must be considered. These cases are:

1. **Vertex-Glue**: An active boundary node shared among blocks
2. **Edge-Glue**: A boundary node active in one block, but absent in the adjoing block
3. **Inactive Node**: An inactive boundary node in one block, but absent in the adjoing block

Vertex-glue is treated in the obvious way – the node is present in both blocks in the final mesh. Edge-glue represents a boundary dendrite. This requires identifying the edge-glue node as a dendrite and indentifying its master nodes to ensure it is properly slaved. During conversion to an unstructured format, treatement of zones containing edge-glue nodes is analogous to that of dendritic zones (see section 2.1) with the added complication that the zones are in different blocks. Inactive boundary nodes are the end points of inactive edges extending to a block boundary. Inactive boundary nodes exist to maintain the logically rectangular structure of the block, but are ignored in block-to-block connectivity calculations.

3 Feathering

Feathering is the process used to convert a structured mesh block into a dendritic mesh block. Currently, feathering is limited to a single logical direction within any given mesh block. Multi-directional feathering capability is in development.

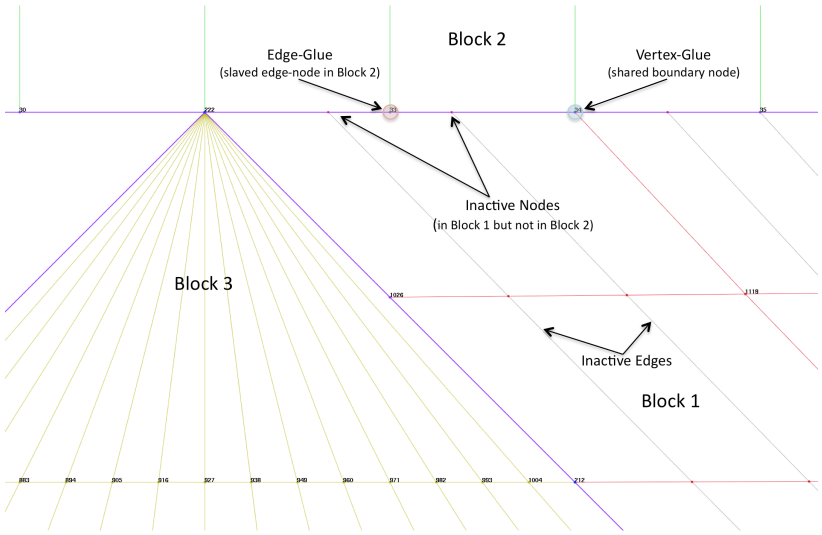


Fig. 6. Block-to-block connectivity: Vertex glue, edge glue, and inactive vertices.

Validity Constraints

A valid dendritic block-structured mesh must obey the following constraints:

1. The set of mesh zones is a disjoint partition of rectangles in the logical space of a mesh block. This basically means that zones must be convex in logical space (e.g., no L-shaped zones) and that there can be no dangling mesh edges (i.e. a vertex must have two active mesh edges).
2. *Slave* nodes cannot also be *Master* nodes (i.e., no recursive slaving)

Base Mesh Resolution

Unlike a structured mesh, a dendritic mesh block does not have constraints on the resolution of its boundaries in the direction of feathering. The boundary resolutions in the logical direction(s) of feathering need not match one another and may each be different than the resolution of the block in logical space. In some cases, the geometry of a mesh block may dictate that the maximum active edge count in the block occur in the interior of the block to maintain a desired zone size (*e.g.,* meshing a football). Figure 7 shows an example of a boundary resolution mismatch and an interior resolution that is larger than either of the boundary resolutions. The mesh block in Figure 7 is generated

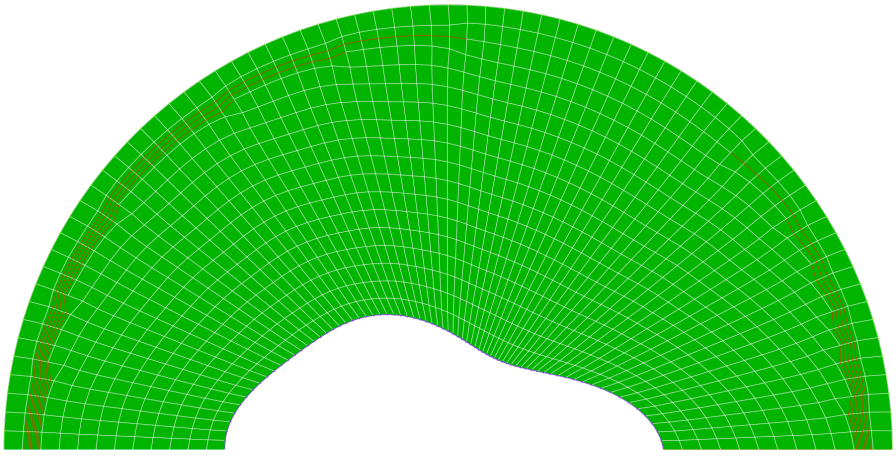


Fig. 7. Feathering with an interior and boundary resolution mismatch (red lines are inactive edges).

with the constraint that a uniform zone size be maintained. Note that for the case in Figure 7 the i-resolution of the block is determined by the resolution needed in the interior of the block to maintain uniform zoning after feathering.

Feather Schemes

Feather schemes determine the order in which mesh edges are removed for a given logical sheet. Note however, that the scheme does not determine *how many* edges are removed. There are three feathering schemes (patterns) currently supported. They are *boundary feathering* where edges are removed near one of the logical boundaries of the block, *centered feathering* where edges are removed near the center of the block logical space, and *distributed feathering* where edges are removed in a distributed manner throught the logical space of the block. The *boundary* and *distributed* schemes have the option to prevent edges from being removed immediately adjacent to a boundary by specifying an integer *offset* to indicate how many layers of un-featherd zones to leave next to either or both boundaries. Figures 7 and 8 show examples of the *boundary* and *centered* schemes. The *distributed* scheme is shown in Section 1, Figure 1.

Distributed feathering requires that edges to be removed from the mesh be more-or-less uniformly distributed across the width of the mesh. We do not want the removed edges to cluster to one side or the other, or to be preferentially located near the block's center or near the block boundaries. This outcome is achieved by means of a repulsive "potential" acting between removed edges. The potential is calculated in logical coordinate (index) space, and periodic boundary conditions are employed to prevent removed edges from clustering near the block boundaries. We arbitrarily use a $1/r^2$ potential (a

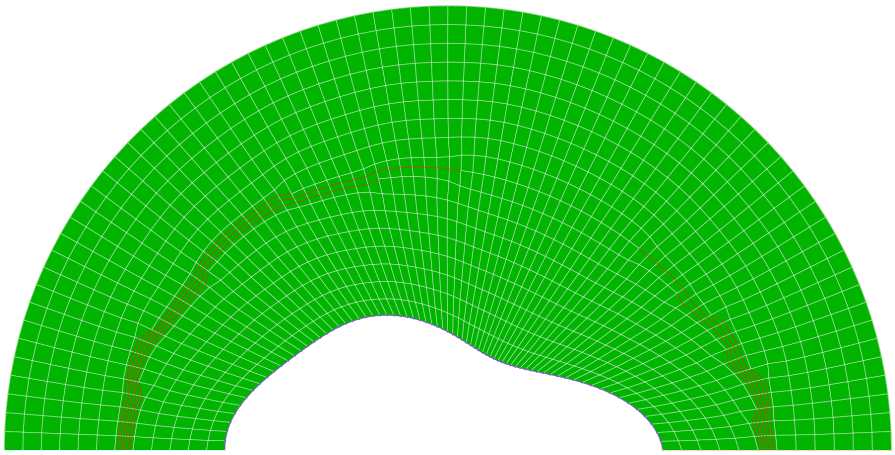


Fig. 8. Feathering with the *center* scheme.

$1/r$ potential would serve equally well) to select the next edge to be removed based on minimization of the potential produced by existing removed edges.

Removing Edges

Given a base mesh resolution and a desired feathering scheme, the only piece of information missing is the number of edges to be removed at each logical sheet in the feather direction. The count of edges to be removed is determined by calculating the arclength of the TFI projector in the logical direction perpendicular to the feather direction then calculating the number of edges which must be removed to give the desired zone size on the sheet. Once the number of edges is determined, the feather scheme determines which edges are removed.

4 Trans-Finite Interpolation with Dendrites

Trans-Finite Interpolation (TFI) is a method of positioning the internal vertices of logically-retangular mesh blocks based on the locations of the block's boundary vertices. Internal-vertex positions are computed as the vector sum of two linear interpolations between opposing-face pairs, minus a bilinear interpolation based on the corner vertices. For the special case of a block with two opposing uniformly-distributed straight-line faces, the bilinear interpolation exactly cancels the linear interpolation between the straight-line faces, leaving the linear interpolation between the remaining two (curvelinear) faces.

The simplest possible TFI uses the interior vertex's normalized logical coordinates ($i = I/(N_I - 1), j = J/(N_J - 1)$) as the interpolation parameters.

Such a TFI will not accurately propagate non-uniform boundary vertex distributions into the interior. This defect may be overcome by replacing normalized logical coordinates (i, j) with normalized curvelinear coordinates (u, v), with u and v linear in arc length from 0 to 1 along opposing mesh block faces; u increases with increasing i, and v increases with j. Curvelinear coordinates for the (I, J)-th interior vertex are given by:

$$u = (1 - v)u_l(I) + vu_h(I) \tag{4}$$

$$v = (1 - u)v_l(J) + uv_h(J) \tag{5}$$

where $u_l(I)$ and $u_h(I)$ are the u coordinates of the I-th vertex on the $v = 0$ and $v = 1$ boundaries, respectively. A similar definition holds for $v_l(J)$ and $v_h(J)$.

To extend TFI to dendrited mesh blocks we introduce Normalized Dendrite-Logical (NDL) coordinates (i_d, j_d). NDL coordinates for the (I, J)-th vertex are calculated using active-vertex counts along the I and J mesh lines:

$$i_d = N_I A_{const-J}(I, J)/A_{const-J}(N_I - 1, J) \tag{6}$$

$$j_d = N_J A_{const-I}(J, I)/A_{const-I}(N_J - 1, I) \tag{7}$$

where $A_{const-J}(I, J)$ is the running count of active vertices along the Jth constant-J mesh line starting with $A_{const-J}(0, J) = 0$. $A_{constI}(J, I)$ is defined similarly. Note that NDL coordinates are normalized to the dimensions of the unfeathered mesh block. NDL coordinates are, strictly speaking, only required for active vertices, but it is useful for purposes of mesh visualization to apply the TFI to inactive vertices as well. For this purpose we linearly interpolate i_d values for succesive inactive vertices on a constant-J line from the i_d values of the bracketing active vertices (and like-wise for j_d values on constant-I lines).

TFI on a dendrited block is acomplished by replacing logical coordinates (I, J) in equations 4 and 5 with NDL coordinates (i_d, j_d). The block-boundary coordinates u_l, u_h, v_l and v_h are now determined by linear interpolation between successive boundary vertices, e.g.:

$$\begin{aligned} f(i_d) &= i_d - \lfloor i_d \rfloor \\ u_{l,\text{dendrite}}(i_d) &= (1 - f(i_d))\ u_l(\lfloor i_d \rfloor) + f(i_d) * u_l(\lceil i_d \rceil). \end{aligned} \tag{8}$$

Figure 9 shows the effect of a dendrite aware TFI on a feathered fan mesh. Note that the active vertices are uniformly distributed along each of the block's interior arcs.

5 Dendritic Mesh Smoothing

Historically, there are many methods proposed for mesh enhancement such as those documented in [5], [9],[11], [13], and [14], for example. The presence of

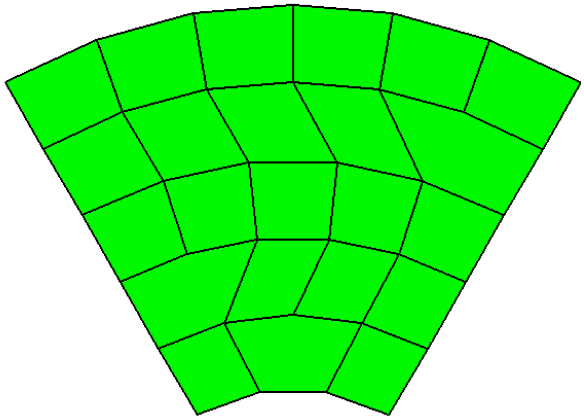


Fig. 9. Application of TFI to a dendritic mesh block.

dendritic zones (elements) within the mesh to be enhanced present challenges to algorithms designed for smoothing non-dendritic meshes. There are two fundamental reasons why this is so: 1.) the presence of extra *hanging* nodes in an element and 2.) applying constraints to those hanging nodes.

In the first case, the presence of hanging nodes means that for the dendritic element the representation of physics is different than it is for the non-dendritic elements. For example, consider a finite element context wherein it is desired to solve a partial differential equation within a domain. Suppose a mesh consists (in two-dimensions) of primarily simplex triangular elements, the presence of a dendritic triangular element in the mesh will cause that element to no longer be a simplex element. It may have up to three mid-side (hanging) nodes. Consequently, dendritic elements are non-conforming (*c.f.,* Section 5.5 of [4]) requiring modifications to the element basis function set.

In the second case, it is often required that the hanging nodes be placed according to a constraint. A common constraint on node j, the local elemental index of a hanging node, in an unstructured mesh is

$$x_j^i = \frac{1}{2}\left(x_{j+1}^i + x_{j-1}^i\right), \tag{9}$$

where x_j^i are the $i = 1, \ldots, D$ coordinate components in D-dimensional space for the $j \in 1, \ldots, N^e$ (counter-clockwise numbered) nodes in element e.

The Laplace-Beltrami method of mesh enhancement (*e.g.,* Chapters 5 – 9, [11]) is used in the following discussion as a means of illustrating how these two issues impact a mesh enhancement algorithm. The discussion draws extensively from Douglass [7].

5.1 Laplace-Beltrami Mesh Enhancement

Consider a domain, Ω, in three-dimensional Euclidean space, $x^i = (x, y),\ i = 1, 2, 3$, which is described locally with parametric coordinates, $u^i = (u, v, w)$. Then, harmonic coordinates, x^i, are defined by the system

$$\frac{1}{\sqrt{g}} \frac{\partial}{\partial u^\alpha} \left(\sqrt{g} g^{\alpha\beta} \frac{\partial x^i}{\partial u^\beta} \right) = 0, \tag{10}$$

where, for the two-dimensional problems discussed here, $i, \alpha, \beta = 1, 2$. The contravariant metric tensor components $g^{\alpha\beta}$ are related to the covariant components through $g^{\alpha\gamma} g_{\gamma\beta} = \delta^\alpha_\beta$ and $g = \det(g_{\alpha\beta})$, where

$$g_{\alpha\beta} = \frac{\partial x^i}{\partial u^\alpha} \frac{\partial x^i}{\partial u^\beta}, \tag{11}$$

and where the usual Einstein summation convention is applied to repeated sub- or superscripts within a term and δ^α_β is the Kronecker delta.

To solve these equations for x^i, the finite element method of weighted residuals (MWR) ([1], [8]) is used to find an approximate solution. The partial differential equation is converted to an equivalent weak form

$$\begin{aligned} I &= \sum_{e=1}^{E} I_e = 0 \\ I_e &= \int_{\Omega_e} \frac{\partial w}{\partial u^\alpha} g^{\alpha\beta} \frac{\partial x^i}{\partial u^\beta}\ \sqrt{g} d^2u - \int_{\partial\Omega_e} w \sqrt{g} g^{\alpha\beta} \frac{\partial x^i}{\partial u^\beta}\ ds_\alpha \end{aligned} \tag{12}$$

where w is an appropriate weight function. The last term,the integral over the boundary of Ω_e, cancels for all interior element boundaries and is taken to be zero for those element boundaries on the domain boundary subject to natural boundary conditions.

The dependent variables within each element use a nodal interpolation of the form

$$x^i = \sum_{k=1}^{N_e} \phi_k(u, v)\ X^i_k \tag{13}$$

where $k = 1, \ldots, N_e$ and N_e is the number of element nodes with $X^i_k = (X_k, Y_k)$ the coordinates of the k^{th} element node and ϕ_k are the N_e basis functions. The element in the physical (x, y)-plane is the image of a unit element in the (u, v)-plane through the mapping $x^i = x^i(u, v)$. It is common to define a basis function set for a master element having coordinates $\xi^i = (\xi, \eta)$ so that the local u^i-coordinates are mapped to the master element via, usually, an isoparametric mapping

$$u^i = \sum_{m=1}^{M} U^i_m\, \psi_m(\xi, \eta), \tag{14}$$

where U^i are the coordinates of the element nodes in u^i-space. Using isoparametric mapping, the number of basis functions in the map is equal to the number of basis functions used in the approximate solution, $M = N_e$. Equation 12 is invariant when the integration variables are changed from u^i to ξ^i resulting in

$$I_e = K^e_{ij} X^i_j, \text{ where} \tag{15}$$

$$K^e_{jk} = \int_{\Omega_e} \frac{\partial \psi_j}{\partial \xi^\alpha} g^{\alpha\beta} \frac{\partial \psi^i_k}{\partial \xi^\beta} \sqrt{g}\, d\xi d\eta. \tag{16}$$

5.1.1 Hanging Nodes and Element Basis Functions

The element basis functions address the first issue discussed above, that of the presence of hanging nodes in dendritic elements. Consider either linear triangles or bilinear quadrilateral elements as in Becker *et al.* [1], for example, whose basis functions are given in the upper part of Table 1.

Table 1. Basis functions for triangle and quadrilateral elements.

Linear Triangle Basis	Bilinear Quadrilateral Basis
$\psi_1 = 1 - \xi - \eta$	$\psi_1 = \frac{1}{4}(1-\xi)(1-\eta)$
$\psi_2 = \xi$	$\psi_2 = \frac{1}{4}(1+\xi)(1-\eta)$
$\psi_3 = \eta$	$\psi_3 = \frac{1}{4}(1+\xi)(1+\eta)$
	$\psi_4 = \frac{1}{4}(1-\xi)(1+\eta)$
6-Node Triangle Basis	**8-Node Quadrilateral Basis**
$\psi_1 = \bar{\psi}_1 - \frac{1}{2}(\Psi_6 + \Psi_4)$	$\psi_1 = \bar{\psi}_1 - \frac{1}{2}(\Psi_8 + \Psi_5)$
$\psi_2 = \Psi_4$	$\psi_2 = \Psi_5$
$\psi_3 = \bar{\psi}_2 - \frac{1}{2}(\Psi_4 + \Psi_5)$	$\psi_3 = \bar{\psi}_2 - \frac{1}{2}(\Psi_5 + \Psi_6)$
$\psi_4 = \Psi_5$	$\psi_4 = \Psi_6$
$\psi_5 = \bar{\psi}_3 - \frac{1}{2}(\Psi_5 + \Psi_6)$	$\psi_5 = \bar{\psi}_3 - \frac{1}{2}(\Psi_6 + \Psi_7)$
$\psi_6 = \Psi_6$	$\psi_6 = \Psi_7$
$\Psi_4 = 4(1-\xi-\eta)\ \xi\ \delta_{s1}$	$\psi_7 = \bar{\psi}_4 - \frac{1}{2}(\Psi_7 + \Psi_8)$
$\Psi_5 = 4\ \xi\ \eta\ \delta_{s2}$	$\psi_8 = \Psi_8$
$\Psi_6 = 4\ \eta\ (1-\xi-\eta)\ \delta_{s3}$	$\Psi_5 = \frac{3}{8}(1-\eta)(1-\xi\eta)\ \delta_{s1}$
	$\Psi_6 = \frac{3}{8}(1+\xi)(1-\eta^2)\ \delta_{s2}$
	$\Psi_7 = \frac{3}{8}(1+\eta)(1-\xi^2)\ \delta_{s3}$
	$\Psi_8 = \frac{3}{8}(1-\xi)(1-\eta^2)\ \delta_{s4}$

Dendritic elements, in the context of the finite element method, cannot be treated as though they are simple 3-node triangles or 4-node quadrilateral elements. The presence of mid-side (*i.e.,* hanging) nodes must be accounted for in the basis function set for the element. Such elements are considered transition elements as in Huang and Xie [12] who present a transition 5-node quadrilateral finite element addressing this refinement issue, which forms the foundation of the treatment of dendritic element interpolation used here. Dendritic triangular elements are handled in an analogous fashion.

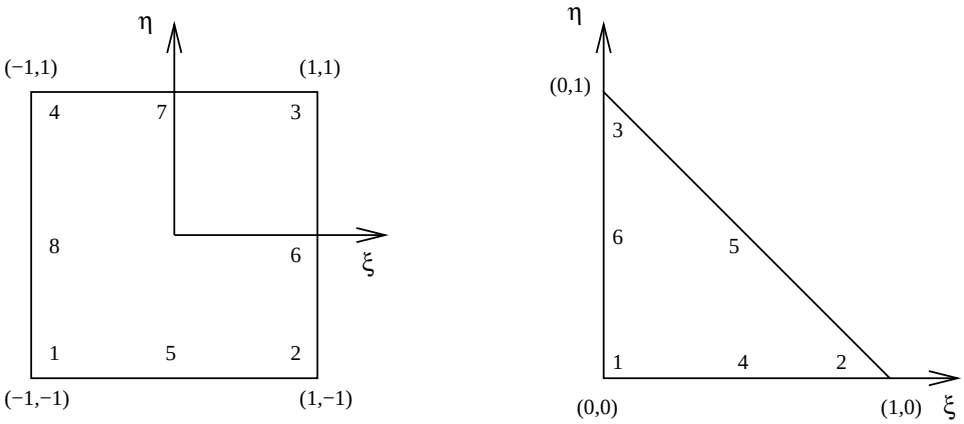


Fig. 10. Master transition quadrilateral (left) and triangular (right) elements.

In general, a transition triangular element may have up to 3 hanging nodes giving as many as 6 nodes per element while a 4-node quadrilateral transition element may have up to 4 hanging nodes leading to as many as 8 nodes in the element. For the most general cases, these elements are shown in Figure 10. If the linear/bilinear basis functions of the upper part of Table 1 are designated with an overbar ($\bar{\psi}_1$, for example), then it can be shown that the basis functions for general transition (*i.e.,* dendritic) elements are as given in the lower part of Table 1. The Ψ_i contain a multiplier, δ_{st}, the Kronecker delta function. The subscripts refer to the sides of the element containing a hanging node.

5.1.2 Hanging Node Constraints in Dendritic Elements

The second issue described above, hanging node constraints, can be addressed in the following manner. We have seen that a typical constraint for hanging node X^i_{HN} in an element is given in Equation 9. This constraint may be imposed by an appropriate penalty method or by explicitly imposing it after each iterative solution of the assembled problem, thereby lagging the solution. In Carey [3], a method is presented which applies a penalty method for inter-element constraints such as for hanging nodes. If the governing system admits

to a variational principle, as does Equation 10, and if the constraint within the element is of the form $\mathbf{G}\,\mathbf{v} = 0$, then the penalty functional becomes

$$J_\epsilon = \frac{1}{2}\mathbf{v}^t\, K\, \mathbf{v} - \mathbf{v}^t\, \mathbf{F} + \frac{\epsilon^{-1}}{2}\mathbf{v}^t\, \mathbf{G}^t \mathbf{G}\, \mathbf{v}, \tag{17}$$

where ϵ^{-1} is the penalty factor and from which the stationary condition gives

$$\left(K + \frac{1}{\epsilon}\mathbf{G}^t\mathbf{G}\right)\mathbf{v} = \mathbf{F}. \tag{18}$$

Since Equation 10 has a variational principle and the constraints are expressible in the form $\mathbf{G}\,\mathbf{v} = 0$, then Equation 18 applies. For an element with N^e nodes the constraint condition vector, $\mathbf{G}$, has components

$$G_i = \begin{cases} 1 & \text{if } i = j \\ -\frac{1}{2} & \text{if } i = j \pm 1 \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

for node j a hanging node. For an element with 5-nodes and node 4 the hanging node, the constraint condition vector would be

$$\left[0, 0, -\frac{1}{2}, 1, -\frac{1}{2}\right] \begin{bmatrix} X_1^i \\ X_2^i \\ X_3^i \\ X_4^i \\ X_5^i \end{bmatrix} = 0, \tag{20}$$

and the element matrix becomes

$$K^e + \frac{1}{\epsilon} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{4} & \frac{-1}{2} & \frac{1}{4} \\ 0 & 0 & \frac{-1}{2} & 1 & \frac{-1}{2} \\ 0 & 0 & \frac{1}{4} & \frac{-1}{2} & \frac{1}{4} \end{bmatrix}. \tag{21}$$

Should there be $N^e_{HN} \leq N^e$ hanging nodes within the element, N^e_{HN} constraint matrices sum together as in Equation 21 for 1 hanging node, using Equation 19 as before.

5.1.3 Metric Tensor Components

I_e is a non-linear function of the nodal coordinates, X^i, through the metric tensor components. In terms of their interpolants, the covariant metric tensor components are

$$g_{11} = \frac{\partial \psi_k}{\partial \xi} \frac{\partial \psi_l}{\partial \xi} X_k^i \, X_l^i, \tag{22}$$

$$g_{12} = \frac{\partial \psi_k}{\partial \xi} \frac{\partial \psi_l}{\partial \eta} X_k^i \, X_l^i, \tag{23}$$

$$g_{22} = \frac{\partial \psi_k}{\partial \eta} \frac{\partial \psi_l}{\partial \eta} X_k^i \, X_l^i. \tag{24}$$

Mesh motion relies upon the metric components to "enhance" the mesh from an initial state to a "better" final state by solving the assembled global problem. The metric components are known only within an element. Consequently, if the metric tensor components are calculated using the initial mesh coordinates, then the metric tensor components are consistent with the initial mesh and the mesh does not "move." To circumvent this problem, a target mesh defined by the centroid of a composite region defined by the elements in contact with a node is used. In general, this point will not coincide with the node's coordinates, but will converge to it in the final solution. For node n, let M elements be in contact with it. Each surrounding element has an area, A_m, and centroidal coordinates, $C_m^i = (X_{C_m}, Y_{C_m})$ (*e.g.,* [2]) so that the centroid coordinates of the composite region are then

$$C_n^i = \frac{\sum_{m=1}^{M} A_m C_m^i}{\sum_{m=1}^{M} A_m}. \tag{25}$$

C_n^i is used in place of X_n^i in the metric tensor component calculations. The centroid of each boundary node that is also a fixed (*i.e.,* a Dirichlet) boundary condition node is set to its current coordinates.

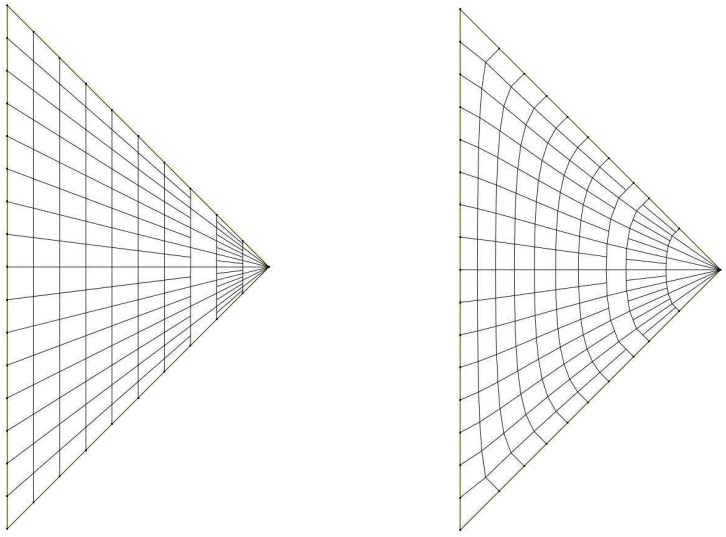
Since the metric tensor components are a nonlinear function of the nodal coordinates, an iterative solution for the coordinates is required. Here a simple fixed point scheme is used, while others have used a Jacobian-free Newton-Krylov method to solve the fully nonlinear problem (*e.g.,* [10]).

5.1.4 Example Results

As an example illustrating enhancement of a composite triangular-quadrilateral element mesh, consider the initial mesh in Figure 11 (left side) having both triangles and quadrilateral elements. The triangle elements are dendritic, while only some of the quadrilateral elements are dendritic. In particular, the quadrilateral dendritic elements have two hanging nodes per element, a more challenging test of the algorithm. This example also illustrates boundary node movement. The right side of the Figure shows the mesh after enhancement, allowing the boundary nodes on the top and bottom boundaries to move, requiring 57 iterations to converge ($\epsilon_{\text{converged}} = 1.\%$).

A mesh quality metric [11](pg. 466) is defined to be E/E_0, where

$$E = \frac{1}{2} \int_{\hat{\Omega}} \left(g^{11} + g^{22} \right) \sqrt{g} \, d\xi d\eta \tag{26}$$



(Initial Mesh) (Enhanced Mesh)

Fig. 11. An example two dimensional initial mesh (left) having both triangular and quadrilateral elements and also "hanging node" or "dendritic" elements. The quadrilateral dendritic elements have two hanging nodes per element and the triangular elements have one hanging node. The enhanced mesh with moving boundary nodes (right) with $E/E_0 = 0.822993$.

and E_0 is E evaluated on the initial mesh. E is the energy functional (variational principle) for the mesh. Consequently, when viewed relative to its initial value it gives a measure of the fractional energy in the mesh with the goal being to minimize the ratio, E/E_0. The final mesh minimizes the energy functional producing a minimum value of E/E_0, giving the solution to Equation 10.

Iterations cease when the largest node movement on the mesh is less than 1% of the maximum node movement of the first iteration.

6 Conclusion

We present a method for constructing dendritic meshes for two-dimensional domains. The resulting meshes consist of predominantly quadrilateral elements with triangles occurring when two quadrilateral nodes occupy the same position, that is, are degenerate quadrilaterals. A detailed discussions of relevant data structure concerns, feathering structured meshes to produce dendritic elements, trans-finite interpolation on dendritic meshes, and smoothing dendritic meshes are presented. Applying these ideas to construct three-

dimensional dendritic meshes is, in principle, straightforward, but has not yet been done.

Acknowledgement. The authors wish to acknowledge these additional members of the ASC Setup Team at Los Alamos National Laboratory without whose contribution this article and the software used for dendritic meshing could not have been completed: S. Davis Herring, Laura M. Lang, and Joseph H. Schmidt.

References

1. E.B. Becker, G. F. Carey, and J.T. Oden. *Finite Elements: An Introduction*, volume I. Prentice-Hall International, London, 1981.
2. Paul Bourke. Calculating the area and centroid of a polygon. http://paulbourke.net/geometry/polyarea, July 1988.
3. G. F. Carey, A. Kabaila, and M. Utku. On penalty methods for interelement constraints. *Computer Methods in Applied Mechanics and Engineering*, 30:151–171, 1982.
4. G. F. Carey and J.T. Oden. *Finite Elements: An Introduction*, volume II. Prentice-Hall International, London, 1983.
5. Graham F. Carey. *Computational Grids: Generation, Adaptation, and Solution Strategies*. Taylor & Francis, Washington, DC, USA, 1997.
6. R. Courant, K. Friedrichs, and H. Lewy. Über die partiellen differenzengleichungen der mathematischen physik. *Mathematische Annalen*, 100(1):32–74, 1928. English translation: IBM Journal, March 1967, 215–234.
7. Rod W. Douglass. Laplace-Beltrami enhancement for unstructured two-dimensional meshes having dendritic elements and boundary node movement. *Journal of Computational and Applied Mathematics*, in review, 2011.
8. Bruce A. Finlayson. *The Method of Weighted Residuals and Variational Principles*. Academic Press, New York, 1972.
9. Pascal Jean Frey and Paul-Louis George. *Mesh Generation: Application to Finite Elements*. Hermes Science Publishing, Oxford UK, 2000.
10. G. Hansen, A. Zardecki, D. Greening, and R. Bos. A finite element method for three-dimensional unstructured grid smoothing. *J. Comput. Phys.*, 202(1):281–297, 2005.
11. Glen A. Hansen, Rod W. Douglass, and Andrew Zardecki. *Mesh Enhancement: Selected Elliptic Methods, Foundations, and Applications*. Imperial College Press, 2005.
12. Feiteng Huang and Xiaoping Xie. A modified nonconforming 5-node quadrilateral transition finite element. *Adv. Appl. Math. Mech.*, 2(6):784–797, 2010.
13. Patrick M. Knupp and Stanly Steinberg. *Fundamentals of Grid Generation*. CRC Press, Boca Raton, Florida USA, 1994.
14. Joe F. Thompson, Bharat K. Soni, and Nigel P. Weatherill. *Handbook of Grid Generation*. CRC Press, Boca Raton, Florida USA, 1999.